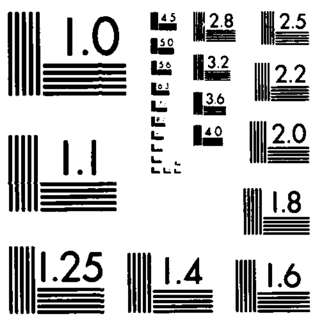
AD-A185 756 LOCAL AND GLOBAL TECHNIQUES FOR THE TRACKING OF PERIODIC SOLUTIONS OF PAR..(U) MINNESOTA UNIV DULUTH DEPT OF MATHEMATICS AND STATISTICS H W STECH 30 APR 87 1/1

UNCLASSIFIED AFOSR-TR-87-1575 $AFOSR-86-0071 F/G 12/1 NL

END
DATE
FILMED
12 87



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED

DTIC FILE COPY

2

AD-A185 756

DTIC ELECTE OCT 2 9 1987 D

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR- 87-1575
6a. NAME OF PERFORMING ORGANIZATION University of Minnesota	6b. OFFICE SYMBOL (If applicable)
7a. NAME OF MONITORING ORGANIZATION AFOSR/NM	
6c. ADDRESS (City, State, and ZIP Code) Duluth Minnesota 55812-2496	7b. ADDRESS (City, State, and ZIP Code) 20332-6448
8a. NAME OF FUNDING / SPONSORING ORGANIZATION AFOSR	8b. OFFICE SYMBOL (If applicable) NM
9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR -86-0071	
8c. ADDRESS (City, State, and ZIP Code) AFOSR/NM Bldg 410 Bolling AFB DC 20332-6448	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
61102F	2304	A9	

11. TITLE (Include Security Classification)
Local And Global Techniques For The Tracking Of Periodic Solutions Of Parameter-Dependent Functional Differential Equations

12. PERSONAL AUTHOR(S)
Dr. Harlan W. Stech

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 3/1/86 TO 4/30/87	4/30/87	

16. SUPPLEMENTARY NOTATION
Supersedes AD-A183222

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This project initiated various aspects of an ongoing study of numerical/analytic techniques for the identification of periodic solutions to functional differential equations. The techniques developed apply ot very general classes of equations, and have been implemented on a variety of specific model problems.

"Local" techniques refer to methods that apply to the problem of analyzing the Hopf bifurcation structure of small periodic orbits of multiparameter systems. A FORTRAN code, BIFDE, was written to analyze generic bifurcations of general systems with infinite delay.

"Global" tracking methods have been developed to study the growth and parameter dependence of global Hopf bifurcations. Investigations have centered on the development of spline-based approximation techniques and their implementation in a FORTRAN code FDETRAK.

Keywords: mathematical programming; machine coding; subroutines; numerical analysis).

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Arje Nachman	22b. TELEPHONE (Include Area Code) (202) 767-5027
22c. OFFICE SYMBOL NM	

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED

Final Technical Report

on

AFOSR-TR- 87-1575

LOCAL AND GLOBAL TECHNIQUES FOR THE TRACKING OF PERIODIC SOLUTIONS OF PARAMETER-DEPENDENT FUNCTIONAL DIFFERENTIAL EQUATIONS

AFOSR-86-0071

March 1, 1986 - April 30, 1987

Dr. Harlan W. Stech
Department of Mathematics
Virginia Polytechnic Institute
and State University
Blacksburg, Virginia 24060

and

Department of Mathematics
and Statistics
University of Minnesota - Duluth
Duluth, Minnesota 55812

Summary

This project initiated various aspects of an ongoing study of numerical/analytic techniques for the identification of periodic solutions to functional differential equations. The techniques developed apply to very general classes of equations, and have been implemented on a variety of specific model problems.

"Local" techniques refer to methods that apply to the problem of analyzing the Hopf bifurcation structure of small periodic orbits of multiparameter systems. A FORTRAN code, BIFDE, was written to analyze generic bifurcations of general systems with infinite delay.

"Global" tracking methods have been developed to study the growth and parameter dependence of global Hopf bifurcations. Investigations have centered on the development of spline-based approximation techniques and their implementation in a FORTRAN code FDETRAK.

Research Objectives

As the beginning of a large on-going project on numerical techniques for the analysis of general parameter-dependent functional differential systems, emphasis was placed on the establishment of the numerical characteristics of some of the proposed methods. An algorithm of the PI for the analysis of the bifurcation structure of Hopf bifurcations chosen to be implemented in a general purpose code BIFDE. A graduate student, Archana Sathaye was supported by the grant to assist in the implementation.

Both Fourier series-based and spline-based methods are potentially valuable in the approximation of the periodic solutions (both large and small) of general functional differential systems. A particular model of neuron firing was chosen to investigate the feasibility of the former, while spline-based methods formed the basis behind a more general purpose FORTRAN code, FDETRAK. Comparison of the two methods, as well as benchmark studies of the latter, formed the bulk of this aspect of the project. A supported graduate student, Toanhung Doanvo, assisted in this portion of the project.

Research Status

Local Analysis: As suggested in the original proposal, a special case of the Lyapunov-Schmidt based stability procedure derived previously by the PI has been successfully implemented in a FORTRAN code BIFDE, whose development, coding and testing constitute the major portion of the Master's Thesis of Archana Sathaye (reference [3] below), completed July, 1986 at the Virginia Polytechnic Institute.

The package is modular, and consists of several routines which perform one or more tasks. In conjunction with the routines available in the package, the user is required to provide a few routines that describe the specific system under study (eg, bifurcation data, characteristic equation, quadratic and cubic nonlinearity). It should be noted that the code now allows numerical analysis of a much wider class of equations than previously possible. The code has been written for VAX computers, and suggests no difficulty in the analysis of small to moderate sized systems, given that one can obtain the prerequisite data called for by the program.

The code has been extensively tested on models from the areas of mathematical epidemiology and genetic repression. In the first case, this numerical confirmed an analysis of the PI that treated a cyclic epidemic model in the setting of functional differential equations with infinite delay. The code treats the system as a system of differential equations, a more natural setting.

A class of genetic repression models of Mahaffy was considered next. Application of the code to this delay difference model was

direct, and allowed a substantial enlargement of the scope of the numerical analysis over that originally done.

A third application was made to a model of "chugging" in liquid propellant rockets. An analysis of the derivation of the linear model of Tischer and Bellman revealed nonlinearities to be attributed to viscous friction forces and fluid flow through the fuel injection nozzle (Torricelli's law). These nonlinearities were retained, and the resulting model was analyzed with the aid if BIFDE. No previous attempts had been made to analyze the structure of self-oscillations in such a nonlinear model. Our work suggested the existence of unstable periodic orbits, which would be consistent with experimental work cited in the literature.

Additional details are available in the cited thesis, which also contains a code listing. With the completion of additional testing, results of this aspect of the project will be prepared for publication.

Global Analysis: Numerical tracking methods have been the stress of a sizeable portion of the project. The long term goal is to develop a portable automatic code for the tracking of Hopf bifurcations in single-parameter systems, with identification of stability and secondary bifurcations, as well as their evolutions.

Work with A. Castelfranco has been completed on the application of Fourier series approximation techniques to a model of delayed feedback in neural systems. Such methods, while accurate and efficient in handling periodic solutions near point of Hopf bifurcation, are not particularly suited for accurate approximation of large periodic orbits. This observation motivates an alternate spline based approximation scheme.

The PI has developed a working partially automatic curve tracking code FDETRAK for the implementation of such a scheme. As is the case of the code BIFDE, on input one must provide the specifics of the model under study (eg, bifurcation data, linearizations). The code automatically selects stepsize strategies to continue the one-parameter family of orbits. Floquet multipliers are computed by approximating the Poincare map associated with the periodic orbits. In particular, a finite dimensional approximation of the phase space leads to a finite dimensional approximation of the period map. Multipliers can then be approximated with the aid of standard eigenvalue packages (eg, IMSL).

Currently, the code is designed to analyze a restricted class of one and two-dimensional scalar delay difference equations. Stabilities of periodic orbits are computed, and secondary bifurcations from the primary branch are identified. Code parameters set a variety of algorithmic variables such as spline order, grid density, dimension of approximating phase space, step-size criteria, frequency of multiplier calculation, stopping criteria, etc.

Initial work in the Virginia Tech VAX 11/785 has shown this

approach to be beyond the capabilities of such machines. Supercomputer time (Cray-2) was obtained from the Minnesota Supercomputer Institute, and basic benchmarks were performed. A 30:1 improvement in running time was observed in comparison to the same (unvectorized) code on the University of Minnesota - Duluth VAX 750. Such results point to the need for the identification of reliable, yet less time-consuming algorithms, as well as the investigation of parallel algorithms. Supported graduate student Tuanhung Doanvo (Virginia Tech) has concerned the use of subspace iteration methods to speed the multiplier calculations. Work along these lines continues.

As restricted as the code now stands, it has been tested on a variety of delay-difference models, and has provided new information about them. In particular, numerical work appears to confirm a conjecture of Chow and Walther concerning the behavior of a model from nonlinear optics. A epidemic model of Mackey has been considered and the numerical results considered in comparison with the work of Sternberg on the subject. Again, the algorithm is highly reliable, although numerically intensive.

A copy of the code FDETRAK constitutes Appendix I of this report.

Research Publications

1. Periodic solutions in a model of recurrent neural feedback, by A. Castelfranco and H. Stech, SIAM Journal of Applied Mathematics, in press.

2. A numerical analysis of the structure of periodic orbits in autonomous functional differential equations, by H. Stech, to appear in Dynamics in Infinite Dimensional Systems.

3. BIFDE: A numerical software package for the Hopf bifurcation problem in functional differential equations, by Archana Sathaye, Master's Thesis, Virginia Polytechnic Institute, Blacksburg, Virginia

(Six copies of each are submitted with this report.)

Research Personnel

Harlan W. Stech, Principal Investigator

Archana Sathaye, graduate student

Toanhung Doanvo, graduate student

Coupling Activities

Conference Participation:

Conference on the Dynamics of Infinite Dimensional Systems, May, 1986, Lisbon, Portugal.

Annual meeting of the Applied and Computational Mathematics Program (DARPA), October, 1986, Boston, Mass.

Appendix I: FDETRAK

```
c
c       FORTRAN     June 25, 1987
c
c       INPUT FILE:
c               FORT.10         READING CURVE POINTS (OPTIONAL)
c               FORT.12         PARAMETERS
c
c       OUTPUT FILES:
c               FORT.11         COLLOCATION COEFFS FOR GRAPHING, STARTUP
c               FORT.15         MULTIPLIERS AND EVECTORS
c               FORT.17         GENERAL INFORMATION
c               FORT.18         BIFURCATION DIAGRAM, EXECUTION TIME
c               FORT.19         LARGEST FLOQUET MULTIPLIERS
c
c.
      REAL DSMIN,DSOPT,DSMAX,WO,PO,MAXZNRM,TOL,TOL2,R1
      real zz(117),zt(117),s(100),wd,wdold,wdnext,tstart,h
      integer lnt,lda,unit,inttype,ijob
      INTEGER N,IOPT,MAXSTEP,ITMAX,ORDER,M,DIM
      common lnt,zz,zt,h,order,lda,s,wd,wdold,wdnext,tstart,
     &       h
c
      beg=second()
c
      CALL READPAR(N,ORDER,NET,M,TOL,TOL2,R1,DSMIN,DSOPT,DSMAX,
     &   WO,PO,N,IOPT,MAXSTEP,MAXZNRM,UNIT,NEIG,ITMAX,INTERVAL,
     &   inttype,ijob)
c
      DIM=N+2
c
      CALL FORTRAN(N,DSMIN,DSOPT,DSMAX,WO,PO,IOPT,MAXSTEP,MAXZNRM,
     &    NEIG,unit,ITMAX,NET,DIM,TOL,TOL2,R1,interval,inttype,ijob)
c
      end=second()
      time=end-beg
      write(18,1000) time
 1000 format(1x,'execution time = ', f12.8)
c
c
      STOP
      END
c
c
c*****************************************************
c
      SUBROUTINE READPAR(N,ORDER,NET,M,TOL,TOL2,R1,DSMIN,DSOPT,DSMAX,
     &   WO,PO,N,IOPT,MAXSTEP,MAXZNRM,NUNIT,NEIG,ITMAX,INTERVAL,
     &   inttype,ijob)
      REAL DSMIN,DSOPT,DSMAX,WO,PO,MAXZNRM,TOL,TOL2,R1
      INTEGER N,IOPT,MAXSTEP,ITMAX,ORDER,M,DIM,NUNIT
      INTEGER N,NET,NEIG,INTERVAL, inttype,ijob
c
      OPEN(UNIT=12,FILE='fort.12')
      READ(12,1) N,ORDER,NET,M,TOL,TOL2,R1,DSMIN,DSOPT,DSMAX,
     & WO,PO,N,IOPT,MAXSTEP,MAXZNRM,NUNIT,NEIG,ITMAX,INTERVAL
     & ,inttype,ijob
    1 format(4(i3,/),8(f8.4,/),3(i3,/),1(f8.4,/),6(i3,/))
      return
      end
c
c
c****************************************************
c
      SUBROUTINE FORTRAN(N,DSMIN,DSOPT,DSMAX,WO,PO,IOPT,MAXSTEP,
     &        MAXZNRM,NEIG,unit,ITMAX,NET,DIM,TOL,TOL2,R1,interval,
     &        inttype,ijob)
      DIMENSION Z(100),ZOLD(100),C(90),V(100),ZP(100),A(100,100)
      DIMENSION ZC(117),solder(16,100),ZT(117)
      REAL DSMIN,DSOPT,DSMAX,DS,ZNORM,SCALE,WO,PO,W,P,MAXZNRM
      REAL CNORM,TOL,TOL2,R1
      COMPLEX CVALUES(200),evectors(200,200)
      REAL S,WD,WDOLD,WDNEXT,TSTART
      INTEGER ITNUM,IOPT,NSTEP,LDA,N,MAXSTEP,FLAGO,NDIMNR,TOLDAP1
      INTEGER ITMAX,N,ORDER,DIM,NET,LNT,noldc2,unit,flagf,inttype
      INTEGER ijob
      COMMON LNT,ZC,ZT,H,ORDER,LDA,S,WD,WDOLD,WDNEXT,TSTART,H
c
      FLAGO=0
      flagf=0
      LDA=N+2
      toLDAP1=2*LDA+1
      NDIMNR=LDA*(2*LDA + 13)/2
      NOLDC2=0
      NSTEP=-1
      call runinfo(h,n,order,net,dim,r1,tol,tol2,
     &       dsmin,dsopt,dsmax,iopt,unit,itmax,interval,neig,
     &       maxstep,maxznrm)
      DS=DSOPT/2.
      CALL ISDATA(SCALE,DS,N,C,WO,PO,Z,ZOLD,LDA,inttype)
c
   10 NSTEP=NSTEP+1
c
   20 CALL PREDICT(SCALE,DS,Z,ZOLD,ZP,V,LDA)
c
      CALL CORRECT(ITNUM,LDA,N,ZP,Z,V,C,W,P,DS,FLAGO
     &       ,NEIG,unit,ITMAX,NDIMNR,TOLDAP1,SCALE)
c       On exit, zp = last computed value (will be rotated to z)
c                  z is unchanged (will be rotated to zold)
      IF(FLAGO.EQ.1)GO TO 99
      IF(FLAGO.EQ.0)GO TO 50
c       FLAGO.EQ.2
      CALL HALVEDS(DS,DSMIN,FLAGO)
      IF(FLAGO.EQ.1)GO TO 99
      GO TO 20
c
   50 continue
      CALL UPDATA(LDA,Z,ZOLD,ZP,solder,interval)
c
      CALL NORMS(C,N,SCALE,W,P,ZNORM,CNORM,WO,PO)
c
      CALL OUTPUT1(NSTEP,LDA,Z,ITNUM,IOPT,DS,DSOPT,DSMIN,DSMAX,
     &                 FLAGO,ZNORM,CNORM,MAXSTEP,MAXZNRM,ZOLD)
c
      CALL CHOICE(maxznrm,maxstep,nstep,interval,p,w,flagf,flago,ijob)
c
      if(flago.eq.1)go to 99
c
      if(flagf.eq.0)go to 90
      flagf=0
c     default is to avoid multiplier calculation on next pass
c
      CALL FMAP(DIM,NET,TOL,A,FLAGO)
      IF(FLAGO.EQ.1)GO TO 99
c
      CALL MULTS(DIM,A,CVALUES,FLAGO,evectors,ijob)
      IF(FLAGO.EQ.1)GO TO 99
c
      CALL ACCOUNT(DIM,CVALUES,TOL2,R1,NOLDC2,FLAGO,evectors,ijob)
      IF(FLAGO.EQ.1) GO TO 99
c
   90 cALL SETDS(DS,DSMAX,DSMIN,DSOPT,IOPT,ITNUM)
c
      GO TO 10
   99 RETURN
      END
c
c*************************
c
      subroutine choice(maxznrm,maxstep,nstep,interval,p,w,flagf,
     &     flago,ijob)
      integer maxstep,nstep,interval,flagf,flago,ijob
      real p,w,maxznrm
      flagf=1
c     Default is now set to compute multipliers
c
      if((nstep/interval)*interval.ne.nstep)flagf=0
c     multipliers now computed only if nstep is a multiple of interval
c
      if((p.lt.1.90).or.(p.gt.2.05))flagf=0
c
c       exit subroutine with flagf = 0 to avoid multiplier calculation
c       exit subroutine with flago = 1 to terminate program
c       exit with ijob = 0 to calculate evalues only
c       exit with ijob = 1 to also calculate evectors
c
c
      IF(NSTEP.LE.MAXSTEP)GO TO 10
      FLAGO=1
      write(17,1001)
 1001 FORMAT(5X,'PROGRAM TERMINATED WITH NSTEP = MAXSTEP')
      GO TO 99
   10 IF(ZNORM.LE.MAXZNRM)GO TO 99
      FLAGO=1
      write(17,1002)
 1002 FORMAT(5X,'PROGRAM TERMINATED WITH ZNORM > MAXZNRM')
   99 return
      end
c
c****************************
c
      subroutine runinfo(h,n,order,net,dim,r1,tol,tol2,
     &       dsmin,dsopt,dsmax,iopt,unit,itmax,interval,neig,
     &       maxstep,maxznrm)
      real maxznrm
      integer unit, dim, order
      m=dim-2
      write(17,1000)h,n,order,net,m,r1,tol,tol2,
     &       dsmin,dsopt,dsmax,iopt,unit,itmax,interval,neig,
     &       maxstep,maxznrm
 1000 format(2x,'h,n,order', 3i10,
```

```
     &  /,2x,'nst,n',2i10,
     &  /,2x,'rl,tol,tol2',3f12.6,
     &  /,2x,'dmin,dmgt,dmean,iogt,unit,itmax',3f8.4,3i6,
     &  /,2x,'interval,neig,maxstep,maxsarm',3i10,f6.3)
      return
      end
c
c*******************************************
c
C
      SUBROUTINE PMAP(NP1,NET,TOL,A,NFLAGP)
C
      EXTERNAL FTN
      INTEGER LDA,NP1,NET,K,ORDER,N,IND,NFLAGP,LXT
      REAL T,TSTART,V(2),DSTART,DMIN,DMAX,WKAREA(2,12)
      REAL TEND,B,Y(100),UD,UDOLD,UDNEXT,tt(100),cc(100)
      DIMENSION  A(NP1,NP1),C(24),WW(2,9),S(100),XC(117),XT(117)
      COMMON LXT,XC,XT,K,ORDER,LDA,S,UD,UDOLD,UDNEXT,TSTART,H
C
      n=lda-2
      call sett(n,tt)
      do 70 j=1,n
      cc(j)=s(j)
   70 continue
      p=s(n+1)
      w=s(lda)
      call mdata(tt,n,xt,lxt,cc,xc)
      UDOLD=0.00
      UDNEXT = 0.00
      N=NP1-1
      NS=NET*(N-1)
C APPROX. PHASE SPACE R2 X L2(N-1 DIMENSIONAL BASIS)
      H=S(LDA)
      NV=2
      NW=NV
      B= H/FLOAT(NS)
      IND=1
C
C  CONSTRUCTION OF PERIOD MAP
C
      DO 100 J=1,NP1
C  SET INITIAL CONDITION
      DO 90 I=1,NS
      Y(I)=0.
   90 CONTINUE
      V(1)=0.
      V(2)=0.
C
      IF(J.EQ.1)GO TO 94
      IF(J.EQ.NP1)GO TO 92
      I1=NET*(J-2)+1
      I2=I1+NET-1
      DO 91 I=I1,I2
      Y(I)=1.
   91 CONTINUE
      GO TO 95
   92 V(2)=1.
      GO TO 95
   94 V(1)=1.
C  STEP THRU DIFFERENTIAL EQUATION
C
   95 L=NS
      TSTART=0.
      TEND=0.
      NFLAG=0
   96 UD=Y(L)
      IND=1
        IER=0
      TEND=TEND +B
      IF(TEND.LT.1.)GO TO 97
      DEL=(TEND-1.)/B
      DEL=1.-DEL
      TEND=1.
      NFLAG=1
C
   97 CONTINUE
      VOLD=V(1)
C
      CALL DVERK(NV,FTN,TSTART,V,TEND,TOL,IND,C,NW,WW,IER)
C
      IF(IER.EQ.0)GO TO 5
      write(17,1000)IER
 1000 FORMAT(1X,'TERMINATION IN DVERK WITH IER = ',I5)
      NFLAGP=1
      GO TO 99
C
    5 IF(NFLAG.EQ.1)GO TO 80
      UDOLD=Y(L)
      Y(L)=(V(1)+VOLD)/2.
      L=L-1
      IF(L.EQ.0)L=NS
      LM1=L-1
      IF(LM1.EQ.0)LM1=NS
      UDNEXT=Y(LM1)
      GO TO 96
C
C  EVALUATE MAP APPROXIMATION
C
   80 A(1,J)=V(1)
      A(NP1,J)=V(2)
      DO 83 I=2,N
      A(I,J)=0.
      DO 82 II=1,NET
      IF((II.EQ.1).AND.(I.NE.2))A(I,J)=A(I,J)+DEL*Y(L)
      L=L+1
      IF(L.GT.NS)L=1
      A(I,J)=A(I,J)+Y(L)
      IF(II.NE.NET)GO TO 82
      A(I,J)=A(I,J)-DEL*Y(L)
   82 CONTINUE
      IF(I.EQ.2)A(2,J)=A(2,J)+DEL*(V(1)+VOLD)/2.
      A(I,J)=A(I,J)/FLOAT(NET)
   83 CONTINUE
C
  100 CONTINUE
C
   99 RETURN
      END
C
C***********************
C
      SUBROUTINE FTN(NU,T,U,UPRIME)
      DIMENSION XC(117),XT(117)
      REAL W,P,S(100)
      INTEGER K,ORDER,LDA,NU,LXT
      REAL U(2),UPRIME(2),T,B,TSTART,s,ss
     &           ,UD,UDOLD,UDNEXT,UDELAY
      COMMON LXT,xc,XT,K,ORDER ,LDA,S,UD,UDOLD,UDNEXT,TSTART,H
C
      W=S(LDA)
      P=S(LDA-1)
C   EVALUATE PERIODIC SOLUTION AT T AND T - w
      td=t-w
      call peeval(t,td,foft,foftd)
c
c  foft = periodic solution at t
c  foftd = periodic solution at t-w
C  INTERPOLATE ONLY AFTER 1 DELAY INTERVAL
C
       IF(T.LT.H+B*1.1)GO TO 3
       S=(T-TSTART)/B - .50
       SS=S*S
       UDELAY=.50*(SS-S)*UDOLD+(1.0-SS)*UD+.50*(SS+S)*UDNEXT
       GO TO 5
     3 UDELAY=UD
C
C
c   5 UPRIME(1)=-P*UDELAY/W
c      wright's eqn
c   5 uprime(1)=-(p/w)*(u(1)*foftd + (foft+1.)*udelay)
c
c      waterfaucet eqn
c   5 uprime(1)=-(p/w)*cos(foftd)*udelay
c
c     for mackey's equations
    5 x10=foftd**10
        temp=(1.-9.*x10)/((1.+x10)*(1.+x10))
      uprime(1)=(2./w)*(-1.00265*u(1) - p*temp*udelay)
      UPRIME(2)=-10.0*U(2)
c
c
       RETURN
       END
C
C***********
C
       SUBROUTINE MULTS(DIM,A,CVALUES,NFLAGM,zz,ijob)
       INTEGER DIM,NFLAGM,ijob
       REAL A(DIM,DIM) ,RW(400),WR(100)
       COMPLEX CVALUES(200),zz(dim,dim)
       izz=dim
       NFLAGM=0
       IER=0
       CALL EIGRF(A,DIM,DIM,IJOB,CVALUES,ZZ,IZZ,WR,IER)
       IF(IER.EQ. 0)GO TO 99
       write(17,1000)IER
 1000  FORMAT(1X,'PROGRAM TERMINATED IN MULTS WITH IER = ',I10)
       NFLAGM=1
    99 RETURN
       END
```

```
c
c**************
c
      SUBROUTINE ACCOUNT(N,EVALS,TOL,R1,OLDCT2,FLAGA,evectors,ijob)
      INTEGER COUNT1,COUNT2,ijob,FLAGA,OLDCT2
      COMPLEX EVALS(N),ELARGE(100),evectors(n,n)
      REAL EMOD(100),rvectors(100,100),em(100)
      R2=1.-TOL
      FLAGA=0
      COUNT1=0
      COUNT2=0
      ETOL=1.
      DO 100 J=1,N
      EM(J)=CABS(EVALS(J))
      emval=em(j)
      IF(emval.LT.R1)then
       em(j)=-1
       go to 100
       endif
c       count multipliers with modulous .ge. r1
        COUNT1=COUNT1+1
c       save evalue and real parts of evectors, if computed
        ELARGE(COUNT1)=EVALS(J)
        EMOD(COUNT1)=emval
       if(ijob.eq.0)go to 31
       do 30 jj=1,n
      rvectors(jj,count1)=real(evectors(jj,j))
   30 continue
   31 continue
c      count mults with modulous .ge. r2 = 1. - tol
      IF(emval.LT.R2) GO TO 100
        COUNT2=COUNT2+1
c
c        find mult nearest to 1.
        ERROR=CABS(1.-EVALS(J))
        IF(ERROR.LT.ETOL) ETOL=ERROR
  100 CONTINUE
c
c     output
      write(19,1200)(em(jj),jj=1,count1)
      write(17,1201)(em(jj),jj=1,count1)
      do 300 j=1,count1
      write(15,301) elarge(j)
      if(ijob.eq.0)go to 300
      write(15,302) (rvectors(jj,j),jj=1,n)
  300 continue
      write(15,303)
  301 format(2f12.7)
  302 format(5f12.8)
  303 format(1x,'*')
 1200 format(4f12.7)
 1201 format(1x, 'multipliers larger than r1',4f12.7)
      IF(ETOL.GE.TOL) GO TO 500
      IF(COUNT2.NE.OLDCT2) GO TO 600
      GO TO 99
  500 FLAGA=1
      write(17,1500)  ETOL
 1500 FORMAT(1X,'PROGRAM TERMINATED DUE TO ERROR IN UNIT MULTIPLIER',
     &/,1X,'ERROR IN MULTIPLIER IS ',F12.7)
      GO TO 99
  600 CALL OUT2(COUNT2,ELARGE,EMOD)
      OLDCT2=COUNT2
   99 RETURN
      END
c
c**************
c
      SUBROUTINE OUT2(COUNT2,ELARGE,EMOD)
      INTEGER COUNT2,J
      COMPLEX ELARGE(COUNT2)
      REAL EMOD(COUNT2)
      DO 1 J=1,COUNT2
      ER = REAL(ELARGE(J))
      EI = AIMAG(ELARGE(J))
      write(17,1000)  ER,EI,EMOD(J)
    1 CONTINUE
 1000 FORMAT(1X,'MULTIPLIER ',F10.5,' + i* ',F10.5,'  MODULUS',F10.5)
      RETURN
      END
c
c************
c
      SUBROUTINE HALVEDS(DS,DSMIN,FLAGD)
      REAL DS,DSMIN
      INTEGER FLAGD
      DS=DS/2.
      FLAGD=0
      IF(DS.GE.DSMIN)GO TO 99
      write(17,1000)
 1000 FORMAT(1X,'PROGRAM TERMINATED DUE TO LACK OF CONVERGENCE',
     &  /,1X,'OF CORRECTOR UNLESS DS < DSMIN')
      FLAGD=1
   99 RETURN
      END
c
c***********
c
      SUBROUTINE CORRECT(ITNUM,LDA,N,Z,SOLD,V,C,W,P,DS,FLAGC
     &            ,NSIG,unit,ITMAX,NDIMWK,TOLDAP1,SCALE)
      external fcn
      INTEGER ITNUM,LDA,N,J,ITMAX,NSIG,NDIMWK,TOLDAP1,UNIT
      INTEGER IER,FLAGC,NDIMPAR,TOLDAM1,LDAM1
      INTEGER K,ORDER,LXT
      REAL Z(100),SOLD(100),V(100),C(N),W,P,DS,FNORM
      REAL PAR(200),WK(10000),SCALE,XC(117),XT(117)
      COMMON LXT,XC,XT,K,ORDER
c
c     on input, z is the predicted value (called zp in FDETRAN)
c     on input, sold is the current value (called z in FDETRAN)
c     on output, z is the corrected value
c     on output, sold is unchanged
c
      NDIMPAR=2*LDA+2
      LDAM1=LDA - 1
c                 DEFINE PAR (Parameters to be sent to IMSL routine ZSPOW)
      DO 10 J=1,LDA
      PAR(J)=V(J)
      PAR(J+LDA)=SOLD(J)
   10 CONTINUE
      PAR(TOLDAP1)=DS
      PAR(NDIMPAR)=SCALE
c
C PAR(1 THRU LDA) = V
C PAR(LDA+1 THRU 2*LDA) = SOLD = (C, P, W)OLD
C PAR(2*LDA+1) = DS
C PAR(2*LDA+2) = SCALE
c
      FLAGC=0
      ITNUM=0
    1 ITNUM=ITNUM+unit
      IF(ITNUM.GT.ITMAX)GO TO 100
      CALL ZSPOW(FCN,NSIG,LDA,UNIT,PAR,Z,FNORM,WK,IER)
c
C  IER=131 IN IMSL ZSPOW SUGGESTS POOR PROGRESS OF CONVERGENCE
      IF(IER.EQ.131)GO TO 100
C  IER=130 IN IMSL ZSPOW SUGGESTS NSIG TOO LARGE
      IF(IER.EQ.130)GO TO 200
C  IER=129 IN IMSL ZSPOW MEANS ROOT NOT YET FOUND
      IF(IER.EQ.129)GO TO 1
C  IER=0 (THE ONLY REMAINING CASE) INDICATES CONVERGENCE
C
C  DECOMPOSE Z INTO COMPONENTS
      DO 20 J=1,N
      C(J)=Z(J)
   20 CONTINUE
      P=Z(N+1)
      W=Z(N+2)
      GO TO 99
C
C
  100 FLAGC=2
C  HALVE DS;          IF DS > DSMIN THEN RECOMPUTE ZP AND TRY AGAIN
      write(17,1100)
 1100 FORMAT(1X,'FLAGC=2')
      GO TO 99
C
C
  200 FLAGC=1
C  TERMINATE PROGRAM IF IER = 130
      write(17,1200)IER
 1200 FORMAT(1X,'TERMINATION DUE TO FAILURE IN ZSPOW; IER = ',I6,
     &     /,5X,'THIS SUGGESTS NSIG TOO LARGE')
C
   99 RETURN
      END
c
c*********
c
      SUBROUTINE FCN(X,F,LDA,PAR)
      REAL X(LDA),F(LDA),PAR(1),F1,F2,XC(117),XT(117)
      INTEGER LDA,K,ORDER,LXT
      COMMON LXT,xc,xt,K,ORDER
c
c          STEPSIZE CONSTRAINT
      CALL STEPCONS(LDA,X,PAR,F1)
      F(1)=F1
c
c          PHASE CONSTRAINT
      CALL PHASECON(LDA,X,PAR,F2)
      F(2)=F2
```

```
c
c          COLLOCATION EQUATIONS
      CALL COLLOC(Z,F,LDA,N,ORDER,LNT,XT,XC)
c
      RETURN
      END
c
c*********
c
      SUBROUTINE PEANOCON(LDA,Z,PAR,F2)
      INTEGER LDA,n,nm1,j
      REAL F2,Z(LDA),PAR(1)
      real cnorm,scale,c(100)
c
      f2=z(1)
c   impose the constraint that z(1) = 0.
c
      scale=par(2*lda+2)
      n=lda-2
      do  20 j=1,n
      c(j)=par(lda+j)
   20 continue
      call cnorm(n,scale,c,cnorm)
      if (cnorm .lt.2.) go to 99
cccccc previous line essentially impose z(1) =0.
      nm1=n-1
      f2=0.
      do 100 j=1,nm1
      f2 = f2 + z(j)*(par(lda+j+1) - par(lda+j))
  100 continue
      f2 = f2 + z(n)*(par(lda + 1) - par(lda + n))
      f2 = f2*float(n)*par(2*lda)
c
c     essentially impose minimum movement of coeff vectors
c
   99 RETURN
      END
c
c************
c
      SUBROUTINE COLLOC(Z,F,LDA,N,ORDER,LNT,XT,XC)
      DIMENSION T(100),XT(117),DXC(100,3),C(100),XC(117)
      REAL Z(LDA),F(LDA)
      INTEGER LDA,J,LDAM1
C  N=ORDER OF SPLINES
C  XT= EXTENDED ARRAY OF KNOT VALUES (OF LENGTH LNT=N+2K-1)
C  XC= EXTENDED VECTOR OF SPLINE COEFFICIENTS (OF LENGTH LNT)
C  NEQNS=DIMENSION OF VECTOR-VALUED FUNCTION
C  LTPNM1=N+K-1
C  NTIMES2=2K
C  N=NUMBER OF KNOTS IN (0.,1.)=LT
C  LDA=NEQNS*N+2=DIMENSION OF CURF MATRIX A
C  C=VECTOR OF SPLINE COEFFICIENTS(OF LENGTH N)
C  NDELAY=NUMBER OF DELAYS+1
      REAL P,W,FVALUE
      INTEGER ORDER,K
      LDAM1=LDA-1
      N=LDA-2
      LNT=N+2*K-1
      ND1RP1=ORDER+1
      NEQNS=1
      LTPNM1=N+K-1
      CALL SETT(N,T)
      CALL SETC(N,C,W,P,Z,LDA)
      CALL XDATA(T,N,XT,LNT,C,XC)
      CALL DCOEFF(XT,XC,LNT,LTPNM1,ND1RP1,DXC,K)
      DO 10 J=1,LDA
      CALL EVALUU(J,NEQNS,XT,XC,LNT,DXC,LTPNM1,ND1RP1,K,T,N,FVALUE,W,P)
      F(J)=FVALUE
   10 CONTINUE
      RETURN
      END
c
c**************
c
      SUBROUTINE STEPCONS(LDA,Z,PAR,F1)
      INTEGER LDA,J,TOLDAP1,N
      REAL F1,DIF,SCALE,Z(LDA),PAR(1)
c
c  STEPSIZE CONSTRAINT:
c
c  CNORM(C-COLD)**2 + (W-WOLD)**2 + (P-POLD)**2 - DS**2  = 0.
c
      F1=0.
      TOLDAP1=2*LDA+1
c
c     compute cnorm difference
      SCALE=PAR(TOLDAP1+1)
      N=LDA-2
      DO 10 J=1,N
      DIF = Z(J)-PAR(LDA+J)
      F1= F1 + DIF*DIF
   10 CONTINUE
      F1=SCALE*SCALE*F1/FLOAT(N)
c
c     compute p difference
      J=N+1
      DIF=Z(J)-PAR(LDA+J)
      F1 = F1 + DIF*DIF
c
c     compute w difference
      J=LDA
      DIF=Z(J)-PAR(LDA+J)
      F1=F1+DIF*DIF
      F1=F1 - PAR(TOLDAP1)*PAR(TOLDAP1)
      RETURN
      END
C
C***********
C
      SUBROUTINE OUTPUT1(NSTEP,LDA,Z,ITNUM,IOPT,DS,DSOPT,DSMIN,DSMAX,
     &                 FLAGO,ZNORM,CNORM,MAXSTEP,MAXZNM,ZOLD)
      DIMENSION Z(LDA),ZOLD(LDA)
      INTEGER LDA,N,ITNUM,IOPT,FLAGO,NSTEP,J,MAXSTEP,NP1
      REAL DS,DSOPT,DSMIN,DSMAX,ZNORM,MAXZNM,CNORM,T
C
C
      NP1=LDA-1
      T=1./Z(LDA)
      write(17,1000)NSTEP,ITNUM,ZNORM,DS,CNORM,T,Z(NP1)
 1000 FORMAT(/,1X,'NSTEP,ITNUM,ZNORM,DS',2I10,2F10.5,/,1X,
     & 'CNORM,PERIO,PARAMETER ,3f10.5,/)
      write(10,1200)z(np1),cnorm
      write(11,1300)(z(j),j=1,lda)
 1300 format(5f12.5)
 1200 format(3f12.8)
C
      RETURN
      END
C
C***********
C
      SUBROUTINE UPDATA(LDA,Z,ZOLD,ZP,solder,interval)
      DIMENSION Z(LDA),ZOLD(LDA),ZP(LDA),solder(interval,lda)
      INTEGER LDA,J,interval
c     rotate older values in solder
      m=interval-1
      mp1=interval
      do 9 n=2,interval
      do 8 j=1,lda
      solder(mp1,j)=solder(m,j)
    8 continue
      m=m-1
      mp1=mp1-1
    9 continue
c
      DO 10 J=1,LDA
      ZOLD(J)=Z(J)
      solder(1,j)=z(j)
      Z(J)=ZP(J)
   10 CONTINUE
c   upon return, (z,solder) consists of zk, k=nstep,nstep-1,...,nstep-8
      RETURN
      END
C
C************
C
      SUBROUTINE INDATA(SCALE,DS,N,C,W0,P0,Z,ZOLD,LDA,intype)
      DIMENSION C(N), Z(LDA), ZOLD(LDA)
      REAL P0,Z,DS,TWOPI,R,C,W0,CNORM ,SCALE
      INTEGER J,N,LDA,intype
      TWOPI=6.283185307
      R=0.
C
C  CALCULATE SCALE FACTOR FOR "COEFFICIENT-INTEGRAL" L2 NORM
      DO 10 J=1,N
      C(J) = SIN(TWOPI*R)
      CNORM=C(J)*C(J) +CNORM
      R=R+1./FLOAT(N)
   10 CONTINUE
C
C  SCALE PICKED SO CNORM(SIN**2(TWOPI* )) IS 1.
C
C                        N
C  CNORM = SCALE*SQRT(((1/N)*SUM(C(J)**2)))
C                       J=1
C
       SCALE=1./SQRT(CNORM/FLOAT(N))
C
C
```

```
      if(intype.ne.1) go to 30
C  INTYPE = 1 CORRESPONDS TO HOPF BIFN; CONSTRUCT INITIAL DATA
C  NORMALIZE C AND S TO CORRECT STEPSIZE
      DO 15 J=1,N
      C(J) =DS*C(J)
      SOLD(J)=0.
      S(J)=C(J)
   15 CONTINUE
      S(N+1) = P0
      S(N+2) = W0
      SOLD(N+1)=P0
      SOLD(N+2)=W0
      go to 99
   30 continue
c     intype .ne. 1 corresponds to running start; read data
      open(unit=10,file='fort.10')
      read(10,1200) (sold(j),j=1,lda)
      read(10,1200) (s(j),j=1,lda)
 1200 format(5f12.5)
   99 RETURN
      END
C
C***********
C
      SUBROUTINE NORMC(N, SCALE,C,CNORM)
c
c     Computes the norm of the periodic solution
c
      DIMENSION C(N)
      REAL SCALE, CNORM
      INTEGER J,N
      CNORM=0.
      DO 10 J=1,N
      CNORM= CNORM + C(J)*C(J)
   10 CONTINUE
      CNORM= SCALE*(SQRT(CNORM/FLOAT(N)))
      RETURN
      END
C
C***********
C
      SUBROUTINE NORME(C,N,SCALE,W,P,ENORM,CNORM,W0,P0)
c
c     Computes the distance from (0.,T0,P0) in
c     (P. Solution)x(Period)x(Parameter) space
c
      DIMENSION C(N)
      REAL ENORM,SCALE,W,P,CNORM
      REAL W0,P0,T,T0
      INTEGER N
C
C  T = 1/W IS THE PERIOD
      T=1./W
      T0=1./W0
      CALL NORMC(N,SCALE,C,CNORM)
      ENORM= SQRT( CNORM*CNORM + (T - T0)*(T - T0) + (P - P0)*(P - P0))
      RETURN
      END
C
C********
C
      SUBROUTINE PREDICT(SCALE,DS,S,SOLD,SP,V,LDA)
      DIMENSION S(LDA),SOLD(LDA),SP(LDA),V(LDA)
      REAL DS,NORM,SCALE
      INTEGER N
      NORM=0.
C
C  CALCULATE TANGENT VECTOR AND NORM
C  IN (P. SOLUTION)X(FREQUENCY W)X(PARAMETER) SPACE.
      N=LDA-2
      DO 10 J=1,N
      V(J)=S(J)-SOLD(J)
      NORM = NORM+ V(J)*V(J)
   10 CONTINUE
      NORM=SCALE*SCALE*(NORM/FLOAT(N))
      J=N+1
      V(J)=S(J)-SOLD(J)
      NORM=NORM+V(J)*V(J)
      J=LDA
      V(J)=S(J)-SOLD(J)
      NORM=NORM+V(J)*V(J)
      NORM= SQRT(NORM)
C
C  NORM= SQRT(CNORM**2 + W**2 +P**2)
C
C  NORMALIZE TANGENT VECTOR TO NORM EQUAL TO 1.
C
      DO 20 J=1,LDA
      V(J)=V(J)/NORM
   20 CONTINUE
C COMPUTE SP
      DO 30 J=1,LDA
      SP(J) = S(J) + DS*V(J)
   30 CONTINUE
      RETURN
      END
C
C***********
C
      SUBROUTINE SETDS(DS,DSMAX,DSMIN,DSOPT,IOPT,ITNUM)
      REAL DS,DSMAX,DSMIN,DSOPT
      INTEGER IOPT,ITNUM
c     DS=AMAX1(DSMIN,AMIN1(DSMAX,DSOPT*FLOAT(IOPT)/FLOAT(ITNUM)))
      ds=.5*(ds+dsopt)*float(iopt)/float(itnum)
      if(ds.gt.dsmax)ds=dsmax
      if(ds.lt.dsmin)ds=dsmin
      RETURN
      END
C
C  ************************************************************
C
      SUBROUTINE SETT(N,T)
      DIMENSION T(N)
      REAL S,DELTAN
      INTEGER N,J
C
      S=0.
      DELTAN=1./FLOAT(N)
      DO 10 J=1,N
      T(J)=S
      S=S+DELTAN
   10 CONTINUE
      RETURN
      END
C
C*************************************************************
C
      SUBROUTINE SETC(N,C,W,P,S,LDA)
      DIMENSION C(N),S(LDA)
      INTEGER J
      REAL W,P,S
C
      DO 10 J=1,N
      C(J)=S(J)
   10 CONTINUE
      W=S(LDA)
      P=S(LDA-1)
      RETURN
      END
C
C*************************************************************
C
      SUBROUTINE EVALUB(I,NEQNS,XT,XC,LXT,DXC,LTPKM1,NDIRP1,K,T,N,
     &          TVALUE,W,P)
C
      DIMENSION XT(LXT),XC(LXT),TSEG(20)
     &             ,BSPLN(10,10),BSPLN1(10,10)
     &             ,DXC(LTPKM1,NDIRP1),T(N)
     &            ,F(5),F1(5)
      INTEGER I,NEQNS,LXT,K,J,II,III,N,IMIN,LEFT,KTIMES2,IOPT
     &            ,IM2,LEFT1,IMIN1,MFLAG,NP1,NP2
      REAL TVAL,W,P,TVAL1,DF1,DF2,DF3,DF4,DF5,DF6
C
C          DETERMINE TVAL, LEFT, AND IMIN
C
      IOPT = 1
      NP1=N+1
      NP2=N+2
      IM2=I-2
      CALL TVALUE(IOPT,T,N,IM2,TVAL,K,IMIN,LEFT)
C
      KTIMES2=K*2
      CALL TKNOTS(XT,LXT,LEFT,KTIMES2,TSEG)
C
      CALL BVAL(TVAL,TSEG,K,KTIMES2,BSPLN)
C
      CALL FVAL(LEFT,K,BSPLN,DXC,NDIRP1,LTPKM1,F)
C
      TVAL1=TVAL-W
    2 IF(TVAL1.GE.0.)GO TO 3
      TVAL1 = TVAL1+1.
      GO TO 2
    3 CONTINUE
C
      CALL INTERV(T,N,TVAL1,LEFT1,MFLAG)
      IMIN1=LEFT1-K+1
C
      CALL TKNOTS(XT,LXT,LEFT1,KTIMES2,TSEG)
C
      CALL BVAL(TVAL1,TSEG,K,KTIMES2,BSPLN1)
```

```
c
      CALL FVAL(LEFT1,K,BSPLN1,DXC,NDIRP1,LTPKM1,F1)
c
c     F(1)=Y(T)
c     F(2)=YPRIME(T)
c     F1(1)=Y(T-W)
c     F1(2)=YPRIME(T-W)
c
c      WRIGHT'S EQUATION:
c      FVALUE = W*F(2) + P*(F(1)+1.)*F1(1)
c
c      waterfaucet equation
c      fvalue = w*f(2) + p*sin(f1(1))
c
c       mackey equation
      af1=abs(f1(1))
      fvalue= w*f(2) + 2.*(1.00263*f(1) + p*f1(1)/(1.+af1**10))
c
      RETURN
      END
c
c*************************************************************************
c
      SUBROUTINE TKNOTS(XT,LXT,LEFT,KTIMES2,TSEG)
c
c     COMPUTES A SEGMENT OF BREAK VALUES OF LENGTH K+1
c
c     T(LEFT-K+1) = TSEG(1)
c
c                       .       . .
c     T(LEFT)        = TSEG(K)
c     T(LEFT+1) = TSEG(K+1)
c
c                       .       . .
c     T(LEFT+K) = TSEG(2K)
c
c     WHERE BOUNDARY INDICIES FOR T ARE DEFINED PERIODICALLY VIA XT
c
      DIMENSION XT(LXT),TSEG(KTIMES2)
      INTEGER LXT,LEFT,KTIMES2,K,JJ,J
c
c     K=KTIMES2/2
c     N=LXT-1
c
      DO 30 J=1,KTIMES2
      JJ=LEFT-1+J
c     IF(JJ.LT.1)GO TO 10
c     IF(JJ.GT.LXT)GO TO 20
      TSEG(J) = XT(JJ)
c     GO TO 30
c  10 TSEG(J) = XT(JJ+N) - 1.
c     GO TO 30
c  20 TSEG(J) = XT(JJ-N) + 1.
   30 CONTINUE
      RETURN
c
      END
c
c*************************************************************************
c
      SUBROUTINE BVAL(X,T,K,KTIMES2,BSPLN)
c
c
c  COMPUTES ARRAY OF THOSE SPLINES OF ORDER K NOT ZERO AT X, AS WELL
c  AS THE PREREQUISITE SPLINES OF ORDER K-1,K-2.  SEE P132 DEBOOR.
c
c  T1 < T2 < ... < TK <= X < T(K+1) < ... < T(2K)
c
c  BSPLN(1, ) CONTAINS B(K,1)  0       0       0       0
c  BSPLN(2, ) CONTAINS B(K-1,2) B(K,2)       0       0       0
c  BSPLN(3, ) CONTAINS B(K-2,3) B(K-1,3) B(K,3) 0      0
c
c  BSPLN(K, ) CONTAINS B(1 ,K) B(2 ,K)        .  .  .       B(K,K)
c
      DIMENSION T(KTIMES2),BSPLN(10,10),DELTAR(20),DELTAL(20),BWK(20)
      REAL  X, SAVED ,TERM
      INTEGER  J,I,K,KTIMES2,JP1
c
      J=1
      BWK(1)=1.
      BSPLN(1,1)=1.
c
      IF(J.GE.K) GO TO 99
c
   20 JP1=J+1
      DELTAR(J)=T(K+J) - X
      DELTAL(J)=X - T(K+1-J)
      SAVED=0.
      DO 26 I=1,J
      denom=deltar(i)+deltal(jp1-i)
c     if(denom.lt..00000)write(19,1000)x,(t(j),j=1,ktimes2)
      term=bwk(i)/denom
c     TERM=BWK(I)/(DELTAR(I)+DELTAL(JP1-I))
      BWK(I)=SAVED+DELTAR(I)*TERM
      SAVED=DELTAL(JP1-I)*TERM
   26 CONTINUE
      BWK(JP1)=SAVED
      J=JP1
      DO 27 I=1,J
      BSPLN(J,I)=BWK(I)
   27 CONTINUE
      IF(J.LT.K) GO TO 20
c
 1000 format(1x,'from bval:x,tseg',f12.6,20(/,5e18.5))
   99 RETURN
      END
c
c *************************************************************************
c
      SUBROUTINE XDATA(T,LT,XT,LXT,C,XC)
c
c     CONSTRUCTS EXTENDED KNOT SEQUENCE XT OF LENGTH
c             LXT = LT + 2K - 1
c
c     AND EXTENDED COEFFICIENT VECTOR XC OF SAME LENGTH
c
      DIMENSION T(LT),XT(LXT),C(LT),XC(LXT)
      INTEGER J,JJ,K,LT,LXT,LTPKM1,LTPK
c
      K=(LXT-LT+1)/2
c
c     DEFINE FIRST K-1 TERMS BY PERIODICITY
c
      KM1=K-1
      DO 1 JJ=1,KM1
      XT(JJ)=T(JJ-KM1+LT)-1.
      XC(JJ)=C(JJ-KM1+LT)
    1 CONTINUE
c
c     COPY T KNOTS
c
      LTPKM1=LT+K-1
      DO 2 JJ=K,LTPKM1
      XT(JJ)=T(JJ-KM1)
      XC(JJ)=C(JJ-KM1)
    2 CONTINUE
c
c     DEFINE LAST K KNOTS BY PERIODICITY
c
      LTPK=LT+K
      DO 3 JJ=LTPK,LXT
      XT(JJ)=T(JJ-KM1-LT)+1.
      XC(JJ)=C(JJ-KM1-LT)
    3 CONTINUE
      RETURN
      END
c
c*************************************************************************
c
      SUBROUTINE TVALUE(IOPT,T,N,I,TVAL,K,IMIN,LEFT)
c
c     GIVEN I, COMPUTES TVAL SO AS TO MAXIMIZE B(I,K) AT TVAL
c     ALSO FINDS ASSOCIATED VALUE OF 'LEFT'
c     AND 'IMIN' SUCH THAT AT MOST B(J,K) IMIN<=J<=IMIN+K ARE NONZERO
c     WHEN EVALUATED AT TVAL
c
      DIMENSION T(N)
      REAL TVAL
      INTEGER K,KD2,I,N,IMIN,LEFT,IOPT
c
c            IOPT = 0 INTENDED FOR CALL FROM SREP
c
      IF (IOPT.EQ.1) GO TO 100
      KD2=K/2
      LEFT = I + KD2
      IF(LEFT.GT.N)LEFT=LEFT-N
      IMIN=I-KD2+1
      IF(KD2*2.NE.K) IMIN=IMIN-1
c
      IF(KD2*2.EQ.K)GO TO 2
c               K ODD
       IF(LEFT.EQ.N)GO TO 1
       TVAL=(T(LEFT)+T(LEFT+1))/2.
       RETURN
    1  TVAL=(T(N)+1.)/2.
       RETURN
c
c               K EVEN
    2 TVAL=T(LEFT)
      RETURN
```

```
C
C          IOPT = 1  INTENDED FOR CALL FROM BVPSLV
C
  100 LEFT=I
      TVAL=T(I)
      INITN=I-K+1
      RETURN
      END
C
C *******************************************************************
C
      SUBROUTINE DCOEFF(XT,XC,LXT,LTPKM1,NDIRP1,SCRTCH,K)
C
      DIMENSION SCRTCH(LTPKM1,NDIRP1),XT(LXT),XC(LXT)
      REAL FKMJ,DIFF
      INTEGER  LXT,LTPKM1,NDIR,NDIRP1,K,I,J,JP1,KMJ
C
C
C     COPY XC INTO THE FIRST COLUMN OF SCRTCH
C
      DO 10 I=1,LTPKM1
      SCRTCH(I,1)=XC(I)
   10 CONTINUE
C
C     COMPUTE COEFFICIENT DIFFERENCES (SEE DE BOOR P139 EQN 11B)
C
      NDIR=NDIRP1-1
      DO 20 J=1,NDIR
       JP1= J + 1
       KMJ = K-J
       FKMJ=FLOAT(KMJ)
        DO 19 I=2,LTPKM1
         DIFF=XT(I+KMJ)-XT(I)
         SCRTCH(I,JP1)=((SCRTCH(I,J)-SCRTCH(I-1,J))/DIFF)*FKMJ
   19 CONTINUE
   20 CONTINUE
      RETURN
      END
C
C*******************************************************************
C
      SUBROUTINE FVAL(LEFT,X,BSPLN,DXC,NDIRP1,LTPKM1,F)
C
C   F(1) CONTAINS F(X)
C   F(2) CONTAINS F'(X)
C   . . .
C   F(NDIRP1) CONTAINS F(NDIR)(X)
C
      DIMENSION F(5),BSPLN(10,10),DXC(LTPKM1,NDIRP1)
      REAL SUM
      INTEGER M,LEFT,K,NDIRP1,LTPKM1,KMM,J,LPMM1,LPMM2
C
C
      DO 10 M=1,NDIRP1
      KMM=K-M
      KMMP1=KMM+1
      LPMM1=LEFT+M-1
      LPMM2=LPMM1-1
      SUM=0.
      DO 5 J=1,KMMP1
      SUM = SUM + DXC(J+LPMM2,M)*BSPLN(KMMP1,J)
    5 CONTINUE
      F(M)=SUM
   10 CONTINUE
      RETURN
      END
C
C*****************************************************************
C
      SUBROUTINE INTERV(XT,LXT,X,LEFT,MFLAG)
C
C  SET UP FOR N = LXT SUBINTERVALS OF EQUAL LENGTH
C
C  X SHOULD BE IN [0., 1.)
C
C  OUTPUT: LEFT, MFLAG
C
      DIMENSION XT(LXT)
      REAL X,FN,EPSILON
      INTEGER LXT,N,LEFT,MFLAG
      N=LXT
      FN=FLOAT(N)
      EPSILON=1.000001
C  EPS      INSURES  T(LEFT).LE.X.LT.T(LEFT+1)
      LEFT=INT(FN*X*EPSILON) + 1
C
      IF(LEFT.LT.1)GO TO 10
      IF(LEFT.GE.LXT)GO TO 20
      MFLAG=0
      RETURN
   10 LEFT=1
      MFLAG=-1
      RETURN
   20 LEFT=LXT
      MFLAG=1
      RETURN
      END
C
C     ******************************************************
c
      subroutine pseval(tnow,tdelay,foft,foftd)
      dimension xt(117),xc(117),tseg(20),bspln(10,10)
      real t,td,foft,foftd
      integer lxt,k,order,lda,ktimes2,n
      common lxt,xc,xt,k,order,lda
c
      n=lda-2
c     write(19,1100)(xc(j),j=1,lxt)
 1100 format(1x,'from pseval: xc',20(/,4e10.5))
c     write(19,1200)k,order,lda
 1200 format(1x,'k,order,lda',5x,3i10)
      t=tnow
      if(t.ge.1.)t=t-1.
      if(t.lt.0.)t=t+1.
      call interv(xt,n,t,left,mflag)
      ktimes2 = k*2
      call tknots(xt,lxt,left,ktimes2,tseg)
c     write(19,1300)t,left,(tseg(j),j=1,n)
 1300 format(1x,'from pseval: t,left,tseg',f12.6,i10,20(/,5f10.5))
      call bval(t,tseg,k,ktimes2,bspln)
c
c evaluation loop
      sum=0.
      do 5 j=1,k
      sum=sum+xc(j+left-1)*bspln(k,j)
    5 continue
      foft=sum
c
      td=tdelay
      if(td.ge.0.)go to 3
      td=td+1.
    3 continue
c
      call interv(xt,n,td,left,mflag)
      call tknots(xt,lxt,left,ktimes2,tseg)
      call bval(td,tseg,k,ktimes2,bspln)
c
c evaluation loop
      sum=0.
      do 6 j=1,k
      sum = sum + xc(j+left-1)*bspln(k,j)
    6 continue
      foftd=sum
c
      return
      end
```

```
5       k = order of spline
2       order = order of fde being solved
3       net = number of subdivisions per step in simulation stage
40          m = number of steps in [-w, 0.]   (dim = m+2)
.0001       tol = tol called for in IMSL DVERK
.1          tol2 = tolerance called for in account
.5         r1 = innermost spectral radius
.0005      dsmin = minimum step size
.0025        sdopt = optimal step size
.005         sdmax = maximum step size
.360        w0 = bifurcation frequency
1.52194     p0 = critical bifurcation parameter
40          n = number of collocation points
4           iopt = optimal number of iterations
2        maxstep = maximun number of curvetracking  steps allowed
2.       maxznrm=max norm of curve in  solutionXperiodXparameter space
2       unit = number if iterations per each call to IMSL ZSPOW
3       nsig = parameter called for in IMSL ZSPOW
10      itmax = max number of iterations per each step
1       interval = span between multiplier calculations
2       intype = 1 if Hopf Bifn; 2 if reading data from fort.10
1       ijob = 0 if cvalues only; ijob = 1 if cvalues + cvectors
```

A NUMERICAL ANALYSIS OF THE STRUCTURE OF PERIODIC ORBITS IN AUTONOMOUS FUNCTIONAL DIFFERENTIAL EQUATIONS

Harlan W. Stech
Department of Mathematics and Statistics
University of Minnesota, Duluth
Duluth, Minnesota 55812
and
Department of Mathematics
Virginia Polytechnic Institute
Blacksburg, Virginia 24060

AFOSR-TR- 87-1575

1. Introduction

Understanding the structure of periodic solutions in nonlinear, autonomous functional differential equations is a problem that often arises when such equations are used in the mathematical modeling of "real-world" phenomena. Knowledge of the existence, stability, and parameter dependence of such periodic solutions provides valuable insight into the general dynamics of the system. Stable steady states and periodic orbits are of particular interest since they correspond to observable states in the system being modeled. However, unstable steady states and periodic orbits are of importance as well since (through variation of parameters in the model) these solutions can themselves change stability and therefore, become "observable".

Numerical simulation of the associated initial value problem often provides evidence of the existence of stable equilibria and stable periodic orbits. However, it is of limited value in the study of unstable solutions.

Linearization provides a straight-forward means of analyzing equilibria and their stability types. A careful study of the associated characteristic equation ideally leads to the identification of the subset of parameter space at which a variation of the system parameters can

induce a qualitative change in the nature of solutions near the equilibrium. Generically, system parameters corresponding to the existence of the characteristic value $\lambda=0$ correspond to branch points of equilibria, while the existence of a complex conjugate characteristic root pairs $\lambda=\pm i\omega$ correspond to the existence of small-amplitude periodic orbits.

At parameter values of this last type (Hopf bifurcation) there is now a straight-forward technique for the determination of the stability and parameter dependence (i.e., direction of bifurcation) of such orbits [7]. Fixing all but one system parameter, it is natural to ask how variation of the remaining parameter effects the periodic orbit, inducing changes of stability and secondary bifurcations. Towards this end, the theory of global Hopf bifurcation is valuable in identifying the available alternatives [2].

This paper concerns the use of numerical methods (other than simple simulation studies) to aid in the analysis of both the local and global natures of periodic solutions to parameter-dependent autonomous periodic orbits. Section 2 discusses a numerical implementation ot the Hopf bifurcation algorithm of [7]. Section 3 outlines the use of numerical tracking techniques to determine certain information concerning the global bifurcation picture in one-parameter problems. The usefulness of such techniques is illustrated in Section 3, where the result of the analysis of a model of nerve firing are described.

2. Local Analysis

Consider the differential equation

$$x'(t) = f(\alpha; x_t) \qquad [2.1]$$

in which it is assumed that $x=0$ is an equilibrium for all values of the

system parameters $\alpha \in R^k$. Somewhat arbitrarily, we have chosen $f : R^k \times C \to R^n$, where $C = C([-r,0], R^n)$ is the usual Banach space of continuous R^n-valued functions on $[-r,0]$; other phases spaces can be used as well. Given adequate smoothness (which we, henceforth, assume without mention), we may expand the right hand side in series form

$$x'(t) = L(\alpha)x_t + H_2(\alpha;x_t,x_t) + H_3(\alpha;x_t,x_t,x_t) \cdots , \qquad [2.2]$$

where $L(\alpha)$ is bounded and linear on C, and $H_2(\alpha;\cdot,\cdot)$ and $H_3(\alpha;\cdot,\cdot,\cdot)$ are, respectively, bounded symmetric bilinear and trilinear forms on C.

The linearized problem reads

$$y'(t) = L(\alpha)y_t, \qquad [2.3]$$

which has exponential solutions $y(t) = \varsigma\, e^{\lambda t}$ if and only if

$$[\lambda I - L(\alpha)\, e^{\lambda \cdot}]\, \varsigma \equiv \Delta(\alpha;\lambda)\varsigma = 0. \qquad [2.4]$$

We assume the existence of a critical parameter $\alpha = \alpha_c$ at which [2.4] has a nontrivial solution (i.e., $\det\Delta(\alpha_c;\lambda) = 0$) with $\lambda = \pm i\omega$ a purely imaginary root pair. Assuming simplicit of the root $i\omega$, it is known that the corresponding characteristic vector ς is uniquely defined up to a scalar multiple. Furthermore, the implicit function theorem shows that there exists a unique smooth family $\lambda(\alpha)$ of characteristic roots defined in a neighborhood of α_c in R^k and satisfying $\lambda(\alpha_c) = i\omega$. For α near α_c, we write $\lambda(\alpha) = \mu(\alpha) + i\omega(\alpha)$ and $\varsigma = \varsigma(\alpha)$. For simplicity, we assume that at α_c there are no other purely imaginary root pairs.

Define $\varsigma^* = \varsigma^*(\alpha) \neq 0$ to be any solution of

$\varsigma^*(\alpha)\Delta(\alpha;\lambda(\alpha))=0$ for α near α_c. For λ near $\lambda(\alpha)$, let $\hat{\varsigma}=\hat{\varsigma}(\alpha;\lambda)\equiv\varsigma^*/[\varsigma^*\Delta'(\alpha;\lambda)\varsigma]$, where $\Delta'=\partial\Delta/\partial\lambda$. By simplicity of the characteristic value $\lambda(\alpha)$ the denominator above is nonzero.

The following theorem, whose proof may be found in [7], reduces the problem of analyzing the existence of small periodic solutions with frequence near ω to that of considering a scalar "bifurcation function".

Theorem 2.1: Under the above hypotheses, there are smooth functions $G(\alpha;c,\nu)$ (C−valued) and $x(t,\alpha;c,\nu)$ (R^n−valued and $2\pi/\nu$−periodic in t) defined in a neighborhood of $(\alpha_c,0,\omega)$ in $R^k\times R\times R$ such that [2.1] has a small $2\pi/\nu$−periodic solution $x(t)$ with (α,ν) near (α_c,ω) if and only if $x(t)=x(t,\alpha;c,\nu)$ up to phase shift, and (α,c,ν) solves the bifurcation equation

$$G(\alpha;\ c,\ \nu)=0. \tag{2.5}$$

Moreover,

$$x(t,\alpha;c,\nu)=2\ \mathrm{Re}\{\varsigma(\alpha)e^{\omega i t}\}c+O(c^2), \tag{2.6}$$

G is odd in c, and has the expansion

$$G(\alpha;\ c,\ \nu)=(\lambda-\nu i)c+M_3(\alpha;\ \nu,\ \lambda)c^3+O(c^5), \tag{2.7}$$

where $\lambda=\lambda(\alpha)$, $M_3(\alpha;\nu,\lambda)=\hat{\varsigma}(\alpha;\lambda)\cdot N_3(\alpha;\nu)$,

$$N_3(\alpha;\ \nu)\equiv 3H_3(\varphi^2,\ \overline{\varphi})+2H_2(\overline{\varphi},A_{2,2}e^{2\nu i})+2H_2(\varphi,\ A_{2,0}),$$

with $\varphi(s)=\varsigma(\alpha)e^{i\nu s}$ for $s\le 0$ and $A_{2,2}$, $A_{2,0}$ the unique solutions of

$$\Delta(\alpha;\ 2\nu i)A_{2,2}=H_2(\varphi^2),$$

$$\Delta(\alpha;0)A_{2,0} = 2H_2(\varphi,\overline{\varphi}), \qquad [2.8]$$

respectively.

The imaginary part of [2.5] can be easily solved (e.g., by iteration) to obtain $\nu=\omega(\alpha)+O(c^2)$. Upon substituting this into the real part of [2.5], one obtains the "reduced" bifurcation equation

$$0 = g(\alpha;\, c) \equiv \mu(\alpha)c + K_3(\alpha)c^3 + O(c^5). \qquad [2.9]$$

Given $K_3(\alpha_c)\neq 0$, (the so-called "generic" case), one can show the existence of nonzero solutions $c=c^*(\alpha)$ for values of α near α_c for which $\mathrm{sgn}\{\mu(\alpha)\}=-\mathrm{sgn}\{K_3(\alpha_c)\}$. If in the case $k=1$ $\mu(\alpha)$ increases with α and $K_3(\alpha_c)<0$, the solution of [2.9] near $c=0$ requires $\mu(\alpha)>0$ (supercritical bifurcation). Similarly $K_3(\alpha_c)>0$ corresponds to subcritical bifurcation.

Concerning the stability of the associated periodic orbits, it is known [7] that if all other characteristic roots have negative real parts, then the periodic orbits posses the same stability type as that of c^* when viewed as an equilibrium solution of the scalar ordinary differential equation

$$c' = g(\alpha;\, c) \qquad [2.10]$$

Thus, $K_3(\alpha_c) < 0$ corresponds to an orbitally asymptotically stable periodic orbit, while $K_3(\alpha_c)>0$ corresponds to an unstable periodic orbit.

The above algorithm, although usually too involved to allow an algebraic determination of the structure of Hopf bifurcations, does lend itself to numerical evaluation. This has been recently implemented in the FORTRAN code BIFDE by A. Sathaye [6]. It is there presumed

that one can obtain from the linearized equation [2.3] analytic expressions for $\Delta(\alpha;\lambda)$, as well as the partial derivatives of $\Delta(\alpha;\lambda)$ with respect to λ and some (use-chosen) coordinate of α. It is also assumed that one is able to identify critical values of the parameter α_c for which a simple purely imaginary root pair $\lambda = \pm\omega i$ exists, and the other spectral assumptions listed above hold. Finally, it is expected that $H_2(\alpha;\varphi_1,\varphi_2)$ and $H_3(\alpha;\varphi_1,\varphi_2,\varphi_3)$ can be evaluated, where each of the arguments φ_j are of the form $\varphi_j(s) = we^{zs}$ for complex values z and complex n-vectors w.

Given the above data, BIFDE coordinates the calculation of the left and right characteristic vectors ς^* and ς (by inverse iteration), identification and solution of the linear systems [2.8] (by Gauss elimination with implicit pivoting) and the evaluation of N_3 (hence, M_3 and K_3). The program uses the partial derivatives of $\Delta(\alpha;\lambda)$ to compute $\mu'(\alpha_c)$ the partial derivative of μ with respect to a user-chosen coordinate of α, and thereby determine the direction of bifurcation with respect to that coordinate of α.

The program is complementary to BIFDD of Hassard [5] in that BIFDD assumes [2.1] to be of delay-difference form, yet identifies the required higher order terms numerically. In [5], rather than making use of Theorem 2.1, the stability and direction of bifurcation is determined by center manifold approximation techniques and the Poincaré normal form.

Remark 2.2: Given BIFDE or BIFDD, a principle difficulty lies in the determination of the bifurcation data. That is, the critical value(s) of

the system parameters α and the associated frequency ω of bifurcating periodic orbits. For one-parameter problems (k=1), solution of

$$\det \Delta(\alpha;\, i\omega) = 0 \qquad [2.11]$$

can be obtained by standard rootfinding techniques (e.g., Newton on Quasi-Newton methods) provided the size of the system n is not prohibitively large, and a sufficiently accurate approximation to the bifurcation data is known in advance.

For k large, one can seek bifurcation data by considering the associated nonlinear minimization problem:

> Minimize $\left|\Delta(\alpha;i\omega)\right|^2$, subject to the constraint that α and ω lie within a compact interval of ω values and α lies within a compact subset of admissable system parameters.

Precise approximations for this minimization problem, although useful in specifying the above constraints, are not necessary.

For k=2, one expects the underdetermined system [2.11] to have a one-parameter family of bifurcation data. Given one set of bifurcation data (perhaps by considering the above minimization problem) one can apply now standard continuation techniques to identify curves in parameter space (a subset of R^2) along which [2.11] has a solution. Indeed, in many instances, this family of critical values can be parameterized in terms of ω itself. A simple example serves to illustrate the point.

Example 2.3: Consider an ordinary differential equation

$$x'(t) = f(x(t)); \qquad x \in R^n \qquad [2.12]$$

in which one coordinate x_j of x is thought to act as a feedback in one of the n equations in [2.12]. In studying the effects of time delay in

this feedback, one replaces the appropriate term $x_j(t)$ with $x_j(t-r)$. To determine the stabilizing/destabilizing effect on equilibria, one encounters a characteristic equation of the form

$$p(\lambda) + q(\lambda)se^{-r\lambda} = 0, \qquad [2.13]$$

where p and q are polynomials, r corresponds to the length of the time delay, and s represents a measure of the strength and type (positive or negative) of the feedback. One can algebraically solve for r (then s) in terms of λ by considering [2.13] and its conjugate. The details are elementary and omitted. Observe that this provides a convenient reparameterization of [2.1] in terms of λ rather than r and s in which (generically) with $\mathrm{Re}\{\lambda\}=0$, $\mathrm{Im}\{\lambda\}$ determines the location of α_c on the imaginary root curves in R^2, and with $\mathrm{Im}\{\lambda\}$ fixed, $\mathrm{Re}\{\lambda\}$ determines the stability of the equilibrium.

3. Global Analysis

Consider [2.1] in the special case when $k=1$, and suppose that at some critical value of the parameter α_c the equation has been shown to satisfy the hypotheses of Theorem 2.1. Equation [2.6] provides an asymptotic estimate of the resulting one-parameter family of periodic orbits bifurcating from the equilibrium. We discuss in this section numerical methods for continuation of this one-parameter family away from the equilibrium, calculation of the stability of the orbits, and identification of secondary bifurcation points.

The numerical approximation of periodic orbits must not rely on the stability type of the orbits if a complete global bifurcation picture is to be obtained. For that reason, periodic solutions are viewed as solutions of a boundary value problem of the form

$$F(\alpha, T, x_t, \dot{x}_t) = 0 \qquad [3.1]$$

$x(t+1)=x(t)$, where $F : R\times R^+\times C\times C \to R^n$. The independent variable t has been scaled so that T-periodic solutions of [2.1] correspond to 1-periodic solutions of [3.1].

For periodic solutions x(t) of [3.1] we introduce the finite dimensional approximation

$$x^{(N)}(t) = \sum_{j=1}^{N} c_j\ \phi_j(t), \qquad [3.2]$$

where the ϕ_j represent appropriate scalar 1-periodic basis functions and the c_j are in R^n. Both truncated Fourier series and k^{th} order periodic B-splines are examples of approximations of this type. See [1] and [3].

Collocation provides one means of computing the coefficients c_j. That is, for N distinct nodes t_j chosen from [0.,1.) one considers the nN equations

$$F(\alpha, T, x^{(N)}(t_j), \dot{x}^{(N)}(t_j)) = 0 \qquad [3.3]$$

in the nN+2 unknowns c_j; $j=1,...,N$, α, and T.

We adjoin to these equations a scalar phase constraint to remove the indeterminacy due to the fact that the phase shift of any periodic solution of [3.1] is also a periodic solution. In the case of trucated Fourier series, this corresponds to simply setting one of the coordinates of the primary Fourier coefficients equal to zero. However, there are more sophisticated methods of instituting such a constraint.

Finally, we adjoin a scalar equation in order to (in a sense) specify which of the one-parameter family of periodic solutions is to be

computed. More precisely, given that a solution $(\alpha_i, T_i, x^{(N)}_i)$ to [3.3] has been obtained, one seeks a solution $(\alpha_{i+1}, T_{i+1}, x^{(N)}_{i+1})$ that lies a given arclength away.

Having obtained two points on the one-parameter family of periodic orbits, one can linearly extrapolate ("predict") an initial approximation to the next desired member of the family, then iteratively improve that approximation ("correct") by solving the above nN+2 simultaneous nonlinear equations by some Newton-like scheme.

It should be remarked that in the case of ordinary differential equations, the use of B-splines has an advantage over truncated Fourier series in that the Jacobian matrices encountered are sparse; the precise structure being dependent only on the order of the splines in use. For functional differential equations such sparsity is lost, with the structure of the Jacobian dependent on the form of the equation [3.1] as well as the parameters T and α. Despite this fact, splines possess certain numerical characteristics which speak in favor of their use over truncated Fourier series.

Having computed an approximation $x^{(N)}$ to [3.1] at the parameter values α and T, one determines the stability of the orbit (and identifies secondary bifurcation points) by computing approximations to the orbit's Floquet multipliers. For equations with finite delay, some iterate of the (linearized) Poincaré map is compact [4]. The Floquet multipliers are, therefore eigenvalues of finite multiplicity with zero as their only cluster point.

Let $X^{(M)}$ denote an M dimensional approximation to the phase space X in use. We assume $X^{(M)} \subseteq X$ and let $P^{(M)} : X \to X^{(M)}$ denote a projection of X onto $X^{(M)}$. The approximate (linearized) Poincaré map is

defined to be

$$p^{(M)} = P^{(M)} \circ \Pi \Big|_{X^{(M)}}, \qquad [3.4]$$

where Π is the period 1 map defined by the linearized equation associated with [3.1]. The eigenvalues of $p^{(M)}$ serve as approximations to the Floquet multipliers of the periodic solution to [3.1].

Finally, we remark that due to the autonomous nature of [3.1], 1 is always a Floquet multiplier [4]. This fact provides a useful monitor of the overall accuracy of the periodic solution approximation $x^{(N)}$ and the multiplier approximation scheme described above.

4. An Example

We conclude with a brief description of the results of applying the methodology described in the previous sections to a model in physiology. We refer the reader to [1] for details, and seek only to indicate the kinds of information that can be obtained when applying these ideas to a particular mathematical model.

The two dimensional delay-difference system

$$v' = h(v) - w + \mu[v(t - \tau) - v_0]$$

$$w' = \rho[v + a - bw] \qquad [4.1]$$

arises as a model of recurrent neural feedback. Here, $h(v) = v - v^3/3$, $\rho > 0$ is small, $0 < b < 1$, $1 - 2b/3 < a < 1$, and v_0 is the v coordinate of the unique equilibrium (v_0, w_0) that exists for [4.1] when $\mu = 0$. Fixing ρ, a and b, one can consider the associated Hopf bifurcation problem in the two remaining parameters τ and μ, which are restricted to be positive and negative, respectively.

Linearizing about the equilibrium of [4.1], one obtains as

characteristic equation

$$0 = \lambda^2+(\rho b-\sigma)\lambda+\rho(1+b\sigma)-(\mu\lambda+\mu\rho b)e^{-\lambda\tau}, \qquad [4.2]$$

where $\sigma=h'(v_0)$. As indicated in Section 2, one expects for this two parameter problem that there should be curves of critical parameters at which [4.2] possesses purely imaginary root pairs $\lambda=\pm\omega i$. Since [4.2] has the form [2.13], one expects these "imaginary root" curves to be parameterized by frequency ω.

Figure 4.1 shows a few of these curves for ρ=.08, a=.7 and b=.8. Along them one is able to determine the stability-determining constant K_3 by numerically implementing the algorithm discussed in Section 2. Solid lines correspond to $K_3<0$, while dashed lines correspond to $K_3>0$. One can show that at nonintersection points of the imaginary root curves the required spectral hypotheses hold, and that for small μ that all characteristic roots must have negative real parts. Thus, [4.1] supports both stable and unstable periodic orbits.

If one additionally fixes τ, one can apply numerical tracking techniques similar to those discussed in Section 3 to the resulting one-parameter problem. Figure 4.2 depicts the global bifurcation diagram with $\tau=25$. Solid lines indicate stable periodic orbits, while dashed lines correspond to unstable periodic orbits. See [1] for details.

5. Acknowledgement

This work was partially sponsored by the Air Force Office of Sponsored Research under Grant #86-0071.

6. References

1. Castelfranco, A. and H.W. Stech, Periodic solutions in a model of recurrent neural feedback, SIAM Journal of Applied Mathematics, to appear.

2. Chow, S. - N. and J. Mallet-Paret, The Fuller index and global Hopf bifurcation, Journal of Differential Equations, 29(1978), 66-85.

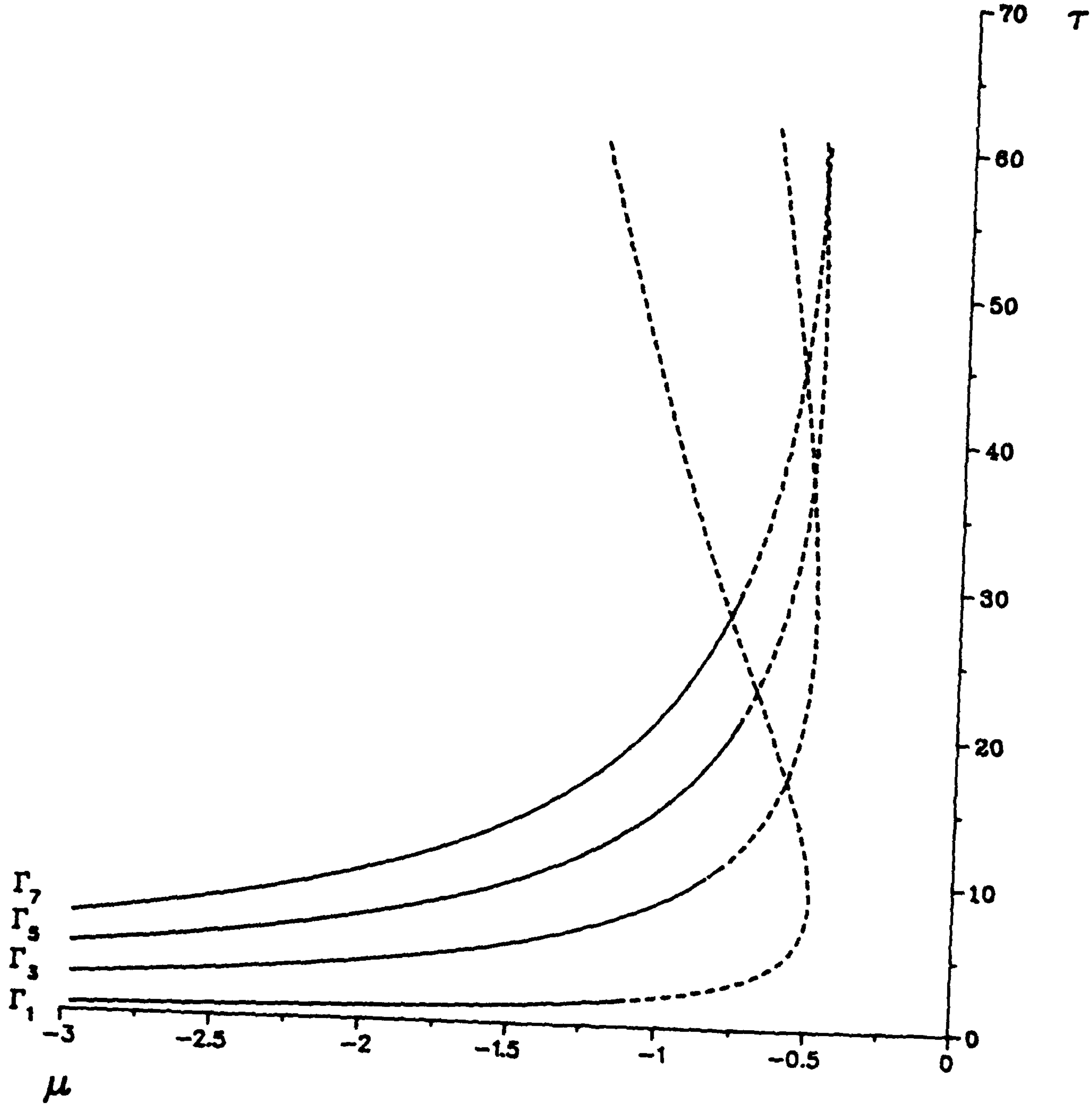
3. Doedel, E.J. and P.C. Leung, Numerical techniques for bifurcation problems in delay equations, Cong. Num., 34(1982), 225-237.

4. Hale, J.K., Functional Differential Equations, Applied Math. Sciences Vol. 3, Springer-Verlag, N.Y., 1971.

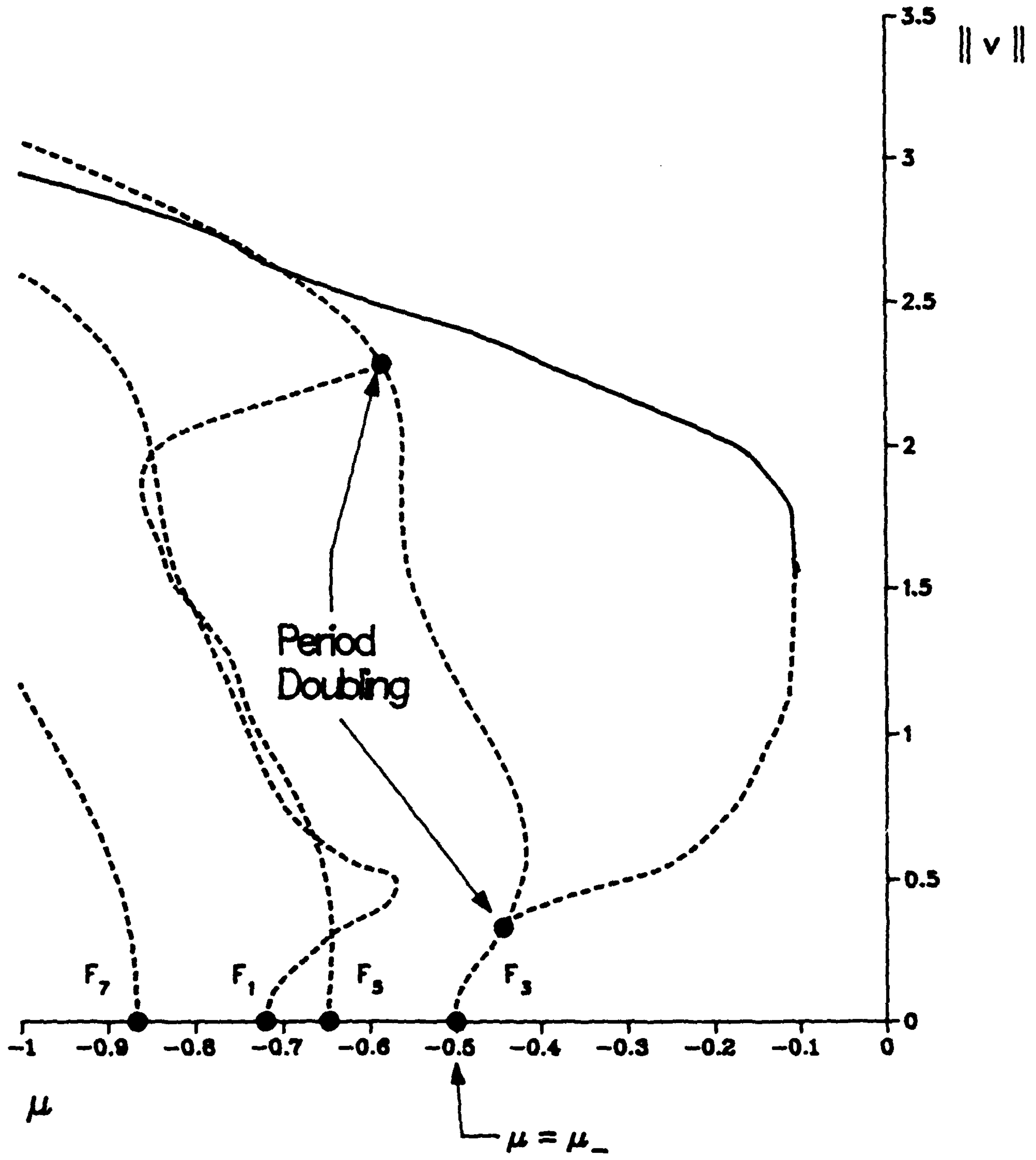
5. Hassard, B., Numerical Hopf Bifurcation Computation for Delay Differential Systems, Proceedings Adelphi University 1982, ed J. P.E. Hodgson, Springer-Verlag Lecture Notes in Biomathematics #51, 1983.

6. Sathaye, A., BIFDE: A Numerical Software package for the Analysis of the Hopf Bifurcation Problem in Functional Differential Equations, Master's Thesis, Virginia Polytechnic Institute, 1986.

7. Stech, H.W., Hopf bifurcation calculations for functional differential equations, Journal of Math. Analysis and Applications, 109(1985), 472-491.



τ
70
60
50
40
30
20
10
0
Γ_7
Γ_5
Γ_3
Γ_1
−3
−2.5
−2
−1.5
−1
−0.5
0
μ



3.5
|| v ||
3
2.5
2
1.5
1
0.5
0
Period Doubling
F_7
F_1
F_5
F_3
-1
-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1
0
μ
$\mu = \mu_-$

DAT

FILM

2